AMENDMENTS TO THE CLAIMS

Claims 1-29 (Cancelled)

30. (New) A machine-readable medium that provides instructions, which when executed by a machine, cause the machine to perform operations comprising:

receiving a binary of a program code, the binary based on a first instruction set architecture;

translating the binary to a translated binary, wherein the translated binary is based at least in part on a second instruction set architecture;

deviating from precise semantics of the binary during said translating in exchange for advantages offered by the second instruction set architecture; and

executing the translated binary.

31. (New) The machine-readable medium of claim 30, further comprising checking settable controls that have been set by a programming environment and that control said deviating.

32. (New) The machine-readable medium of claim 30, wherein said translating comprises storing a portion of a hardware stack in a register of a processor translating the binary.

33. (New) A method comprising:

receiving a binary of a program code, the binary based on a first instruction set architecture;

checking one or more settable controls that have been set by a programming environment to indicate a compatibility level with which to perform the translation;

translating the binary to a translated binary that is based at least in part on a second instruction set architecture, wherein said translating is based at least in part on the controls; and

executing the translated binary.

34. (New) The method of claim 33, wherein the one or more settable controls control a level of semantic compatibility between the binary and the translated binary.

35. (New) The method of claim 33, wherein the one or more controls are set by a programming environment.

36. (New) The method of claim 35, wherein the controls are set by a user.

37. (New) The method of claim 33, wherein the translating and executing are based on a command, the one or more controls based on one or more command line flags associated with the command.

38. (New) The method of claim 33, wherein a register in a processor translating the binary is to store one or more flags corresponding to the one or more controls.

39. (New) The method of claim 33, wherein the first instruction set architecture includes in-order accesses to memory and the second instruction set architecture includes out-of-order accesses to memory, the translating of the binary to include out-of-order accesses to memory by a processor executing the binary.

40. (New) The method of claim 33, wherein the first instruction set architecture allows for self-modifying code and the second instruction set architecture does not allow for self-modifying code, the translating of the binary to include an instruction to controllers of memories that store the binary to perform write operations independent of checks of whether the write operations modify a location where the binary is stored.

41. (New) The method of claim 33, wherein the second instruction set architecture has an address space that is larger than the first instruction set architecture, and wherein the translating of the binary comprises using the address space of the second instruction set architecture.

42. (New) The method of claim 33, wherein data accessed by the binary is stored in a single segment in memory, and wherein an offset value for translating a virtual address to a physical address for the data is not modified during execution of the binary.

43. (New) A system comprising:

a dynamic random access memory to include a binary of a program code based on a first instruction set architecture;

a processor coupled to the dynamic random access memory to translate the binary to a translated binary that is based at least in part on a second instruction set architecture, wherein the processor is to translate the binary by deviating from precise semantics of the binary in exchange for advantages offered by the second instruction set architecture.

44. (New) The system of claim 43, wherein the processor is to deviate based on settable controls that have been set by a programming environment to control the deviation.

45. (New) The system of claim 43, wherein the processor comprises a register to store flags, the flags to indicate at least one translation of a portion of the binary.

46. (New) The system of claim 45, wherein the flags are set by a programming environment.

47. (New) The system of claim 43, wherein the second instruction set architecture has an address space that is larger than the first instruction set architecture, and wherein the translating of the binary comprises using the address space of the second instruction set architecture.

48. (New) The system of claim 43, wherein the binary is stored in a single segment in the memory, and wherein an offset value for translating a virtual address to a physical address is not modified during execution of the binary.

49. (New) An apparatus comprising:

at least one register to store at least one settable flag, the at least one settable flag being settable by a program environment to control a compatibility level of a translation of a binary based on a first instruction set architecture to a translated binary based on a second instruction set architecture;

a decoder to receive the binary and to check the at least one settable flag in the at least one register, the decoder to translate the binary based at least in part on the settable flag.

50. (New) The apparatus of claim 49, wherein the at least one settable flag controls a level of semantic compatibility with which to translate the binary.

51. (New) The apparatus of claim 49, wherein the at least one settable flag is set by a programming environment.

52. (New) The apparatus of claim 49, wherein the translating of the binary comprises storing a portion of a hardware stack in a register of the at least one register.

53. (New) The apparatus of claim 49, wherein the apparatus is coupled to a memory to store the binary, wherein the first instruction set architecture allows for self-modifying code and the second instruction set architecture does not allow for self-modifying code, the translating of the binary to include an instruction to a controller of the memory to cause the memory controller not to check whether code is self modifying.

54. (New) The apparatus of claim 49, wherein the second instruction set architecture has an address space that is larger than the first instruction set architecture, and wherein the translating of the binary comprises using the address space of the second instruction set architecture.

55. (New) The apparatus of claim 49, wherein data accessed by the binary is stored in a single segment in memory coupled to the apparatus, and wherein an offset value

for translating a virtual address to a physical address for the data is not modified during execution of the binary.

56. (New) A machine-readable medium that provides instructions, which when executed by a machine, cause the machine to perform operations comprising:

checking one or more settable controls that have been set by a programming environment to indicate a compatibility level with which to perform a translation of a binary based on a first instruction set architecture to a translated binary that is based at least in part on a second instruction set architecture; and

translating the binary to the translated binary, wherein said translating is based at least in part on the one or more settable controls.

57. (New) The machine-readable medium of claim 56, wherein the one or more settable controls control a level of semantic compatibility between the binary and the translated binary.

58. (New) The machine-readable medium of claim 56, wherein the controls are set by a user.

59. (New) The machine-readable medium of claim 56, wherein said checking the controls comprises checking one or more command line flags associated with a command.

60. (New) The machine-readable medium of claim 56, wherein said checking the controls comprises checking one or more registers of a processor.

61. (New) The machine-readable medium of claim 56, wherein the first instruction set architecture allows for self-modifying code and the second instruction set architecture does not allow for self-modifying code, the translating of the binary to include an instruction to a controller of a memory to store the binary not to check whether a write operation modifies a location where the binary is stored.

62. (New) A system comprising:

a dynamic random access memory to store a binary that is based on a first instruction set architecture that allows binaries to self modify;

a memory controller associated with the dynamic random access memory;

a translation logic coupled to the dynamic random access memory to receive the binary, the translation logic is to translate the binary to a translated binary that is based, at least in part, on a second instruction set architecture that does not allow binaries to self modify, during the translation the translating logic to instruct the memory controller to perform write operations without checking whether binary is self modifying, thereby increasing speed of execution of the translated binary.

63. (New) The system of claim 62, wherein the translation logic is to translate the binary based on at least one settable control that is settable by a program environment.

64. (New) The system of claim 62, wherein the at least one settable control controls a level of semantic compatibility between the binary and the translated binary.

65. (New) The system of claim 62, wherein the at least one settable control allows the decoder to deviate from precise semantics of the binary in exchange for advantages offered by the second instruction set architecture.